(4)

Lockheed Missiles & Space Company, Inc.
SUNNYVALE, CALIFORNIA

LMSC-F301964

Discrete–Element Acoustic Analysis Of

Submerged Structures On Concurrent Computers

B. Nour–Omid, Consultant
J. A. DeRuntz Jr., Senior Staff Scientist
M. A. Wright, Senior Scientist
R. H. Calalo, Associate Senior Scientist

March 1989

Volume II of Final Report
Under Contract N00014-87-C-0863

for

Office Of Naval Research
800 N. Quincy Street
Arlington, Virginia 22217-5000

Mechanics & Materials Engineering Laboratory
Dept. 93-30, Bldg. 251
Lockheed Palo Alto Research Laboratory
3251 Hanover St., Palo Alto, Ca. 94304

ABSTRACT

This report documents the suitability of concurrent computation for frequency domain analysis. When the response of a system is required over a wide frequency range, an immediate increase in computation speed can be attained by assigning each frequency in the range of interest to a separate processor of the concurrent computer. The computations associated with each frequency can be carried out independently and therefore simultaneously with the others. Thus, processing speed-up is roughly proportional to the number of processors in the concurrent computer. This approach is used in discrete element computation for steady state response analysis of fluid-structure systems arising in underwater vibration, acoustic radiation and acoustic scattering problems.

A brief description of the theoretical foundations and governing equations for a boundary-element/finite-element approach for the structural response and surface pressure on a vibrating submerged body is given. The computer code SWEEPS, developed for such analysis, is modified and ported to a 32 processor Ncube concurrent computer. The modified program distributes the frequency dependent computations among the processors. The computational results performed on the Ncube demonstrate a linear speedup with the number of processors. Keywords: Spectrum Analysis, Parallel Processing, Computer Programming

INTRODUCTION

This paper describes an approach for concurrent computation of the steady state solution of fluid-structure systems in the frequency domain using the SWEEPS code [1,2]. Such systems arise in underwater vibration, acoustic radiation and acoustic scattering problems, and the particular fluid-structure interaction formulation used here is the Doubly Asymptotic Approximation (DAA) [3] boundary-element method.

Although theoretically exact formulations of the underwater acoustics problem do exist [4-6], the DAA approach uses a fluid mass matrix and a diagonal fluid area matrix that are independent of frequency. This fact leads to a more efficient use of computational resources when performing variable frequency calculations. In contrast, the governing fluid matrices must be reformed in full for every frequency in the exact formulations. In addition, the exact solution can be prone to the well known *critical frequency* problem [4,5] that the DAA approach does not encounter, although the exact method discussed in [6] does avoid the problem. Hence, DAA methods can be used to advantage in underwater acoustics in that an increase in efficiency can offset some loss of accuracy. Moreover, once the frequency independent matrices that contain all the information that is relevant to the problem have been computed, the calculations for each separate frequency can easily be carried out concurrently with the others.

The primary aim of this paper is to demonstrate the computational speed attainable through a concurrent implementation of the frequency domain calculations as used in the boundary-element/finite-element program SWEEPS. In the next section the governing equations are presented for the steady state vibration of a submerged structure excited either by a set of internal forces with the same specified frequency but otherwise arbitrary magnitudes and phases, or, by an infinite train of sinusoidal incident waves emanating from a spherical source with a specified frequency and magnitude. A simple and direct elimination solution is then given for the structural displacement field and the wet surface scattered pressures. This solution process forms the basis of the SWEEPS processor.

The following section briefly discusses the architecture of the Ncube concurrent computer used in this study. The 32 processor Ncube is a local memory hypercube architecture that uses message passing to communicate information among its processors. The above problem is particularly suitable for such computing environments since a small amount of information is required to be transferred among processors.

The final sections describe the problem that was solved on the Ncube, and the timing results obtained. A linear speed-up with the number of processors is observed in the solution time. These results are then compared with those obtained on the Cray X-MP and the VAX-11/785.

GOVERNING EQUATIONS

The interaction equations for a DAA_2 time-harmonic vibration analysis of a submerged,

linear-elastic structure may be written in matrix form as [1,3]

$$\begin{bmatrix} \mathbf{E}_{ss} & \mathbf{E}_{sf} \\ \mathbf{E}_{fs} & \mathbf{E}_{ff} \end{bmatrix} \begin{Bmatrix} \mathbf{x} \\ \mathbf{p}_s \end{Bmatrix} = \begin{Bmatrix} \mathbf{g}_s \\ \mathbf{g}_f \end{Bmatrix}. \tag{1}$$

where

$$\begin{aligned} \mathbf{E}_{ss} &= -\omega^2 \mathbf{M}_s + i\omega \mathbf{C}_s + \mathbf{K}_s, \\ \mathbf{E}_{sf} &= \mathbf{G}\,\mathbf{A}_f, \\ \mathbf{E}_{fs} &= \rho c i\omega^2 (\omega \mathbf{M}_f - i\boldsymbol{\Omega}_f \mathbf{M}_f)\mathbf{G}^T, \\ \mathbf{E}_{ff} &= -\omega^2 \mathbf{M}_f + \rho c (i\omega \mathbf{A}_f + \boldsymbol{\Omega}_f \mathbf{A}_f), \\ \mathbf{g}_s &= \mathbf{f}_s - \mathbf{G}\,\mathbf{A}_f \mathbf{p}_I, \\ \mathbf{g}_f &= \rho c \omega (\omega \mathbf{M}_f - i\boldsymbol{\Omega}_f \mathbf{M}_f)\mathbf{u}_I. \end{aligned} \tag{2}$$

Here $\mathbf{M}_s$, $\mathbf{C}_s$, and $\mathbf{K}_s$ are the structural mass, damping, and stiffness matrices, respectively, $\mathbf{G}$ is the fluid-structure transformation matrix relating fluid node point forces normal to the wet surface of the structure to node point forces in the structural computational system, $\mathbf{A}_f$ is the wet surface fluid element area matrix, $\mathbf{M}_f$ is the wet surface fluid mass matrix, and $\boldsymbol{\Omega}_f$ is the wet surface fluid frequency matrix; $\mathbf{x}$ is the structural displacement vector, $\mathbf{p}_s$ is the wet surface scattered pressure vector, $\mathbf{f}_s$ is an applied structural force vector, $\mathbf{p}_I$ is an incident wave wet surface pressure vector, and $\mathbf{u}_I$ is an incident wave wet surface normal velocity vector. (If $\mathbf{p}_I = \mathbf{u}_I = 0$, the scattered pressure $\mathbf{p}_s$ reduces to the radiated pressure $\mathbf{p}_R$.) In addition, ρ and c are the fluid density and sound speed, respectively, $i = \sqrt{-1}$, and ω is the frequency of steady state vibration. A superscript T denotes matrix transposition.

The real, symmetric matrices $\mathbf{M}_s$, $\mathbf{C}_s$, and $\mathbf{K}_s$ can easily be generated by any finite-element structural analysis code, and, in the work reported upon here, STAGS (STress Analysis of General Shells) [7] has been used. The real, diagonal matrix $\mathbf{A}_f$ is trivially obtained, while the real, symmetric matrix $\mathbf{M}_f$ can be computed by the boundary-element method of [8]. These two fluid arrays as well as the real, rectangular transformation matrix $\mathbf{G}$ are already produced by the FLUMAS processor of the USA code [9]. Finally, the real matrix $\boldsymbol{\Omega}_f$ may be obtained from either of two formulations, and is such that the matrix product $\boldsymbol{\Omega}_f \mathbf{A}_f$ is symmetric. The development of [3], which is based upon the method of *fluid boundary modes* [8], gives

$$\boldsymbol{\Omega}_f^m = g\rho c\, \mathbf{A}_f \mathbf{M}_f^{-1}, \tag{3}$$

where g is a scalar parameter that can vary between zero and unity. $g = 0$ reduces (2) to the DAA_1 equations, $g = 1/2$ appears to be best for the infinite cylindrical shell, and $g = 1$ is best for the spherical shell. On the other hand, the formulation of [10] does *not* contain any arbitrary parameters as in (3). It is based upon the method of *matched asymptotic expansions*, and, for the fitting procedure described in [11], yields

$$\boldsymbol{\Omega}_f^c = \rho c\, \mathbf{A}_f \mathbf{M}_f^{-1} - c\mathcal{K}, \tag{4}$$

where K is a diagonal matrix of wet surface mean local curvatures. It should be noted that both (3) and (4) do *not* involve any additional information that is not already provided by the FLUMAS processor, in particular, the mean local curvatures are used in the computation of $\mathbf{M}_f$ [8].

Perhaps the most important characteristic of (2) is that the matrices $\mathbf{M}_s$, $\mathbf{C}_s$, $\mathbf{K}_s$, $\mathbf{G}$, $\mathbf{A}_f$, $\mathbf{M}_f$, and $\mathbf{\Omega}_f$ are *frequency-independent*, so that they need only be computed *once* for a complete set of frequency-sweep calculations. This characteristic also renders (1) particularly amenable to incremental iterative methods of solution thus avoiding costly refactorization of the coefficient matrices at every frequency step.

To complete the governing equation system, the form of the right hand side forcing vectors in (2) must also be specified. The elements of the internal forcing vector $\mathbf{f}_s$ can be written as

$$f_{si} = F_i e^{-i\theta_i}, \tag{5}$$

where F_i and θ_i are the magnitude and phase angle respectively of the i^{th} degree of freedom of $\mathbf{f}_s$. Also, the elements of the external forcing vectors $\mathbf{p}_I$ and $\mathbf{u}_I$ can be given for a train of spherical incident waves as

$$\begin{aligned} p_{Ii} &= p_0 \frac{S}{R_i} e^{-ik(R_i - S)}, \\ u_{Ii} &= \frac{p_{Ii}}{\rho c}(1 - i/kR_i)\gamma_i. \end{aligned} \tag{6}$$

Here S is the *standoff*, i.e., the distance between the origin of the spherical wave and the nearest point on the wet surface of the structure, R_i is the distance from the origin of the spherical wave to the i^{th} fluid node on the wet surface, and, γ_i is the cosine of the angle between the vector corresponding to R_i and the wet surface outward normal vector at the i^{th} fluid node. p_0 is the amplitude of the incident pressure at the standoff distance, and k is the wave number ω/c.

Now that the governing equation system for the wet surface unknowns has been fully defined, equations (1) are rewritten by solving the first for $\mathbf{x}$ and substituting into the second. In combination with the first of (1), these become

$$\begin{aligned} (\mathbf{E}_{ff} - \mathbf{E}_{fs}\mathbf{E}_{ss}^{-1}\mathbf{E}_{sf})\mathbf{p}_S &= \mathbf{g}_f - \mathbf{E}_{fs}\mathbf{E}_{ss}^{-1}\mathbf{g}_s, \\ \mathbf{E}_{ss}\mathbf{x} &= \mathbf{g}_s - \mathbf{E}_{sf}\mathbf{p}_S. \end{aligned} \tag{7}$$

$\mathbf{p}_S$ is found from the first of (7) while $\mathbf{x}$ is then obtained from the second. This is the solution procedure currently implemented in the SWEEPS processor.

COMPUTER ARCHITECTURE

The computer used in this study is an Ncube/seven hypercube and consists of 32 (2^5) processors (or nodes), configured as a 5-dimensional hypercube. Figure 1 presents a

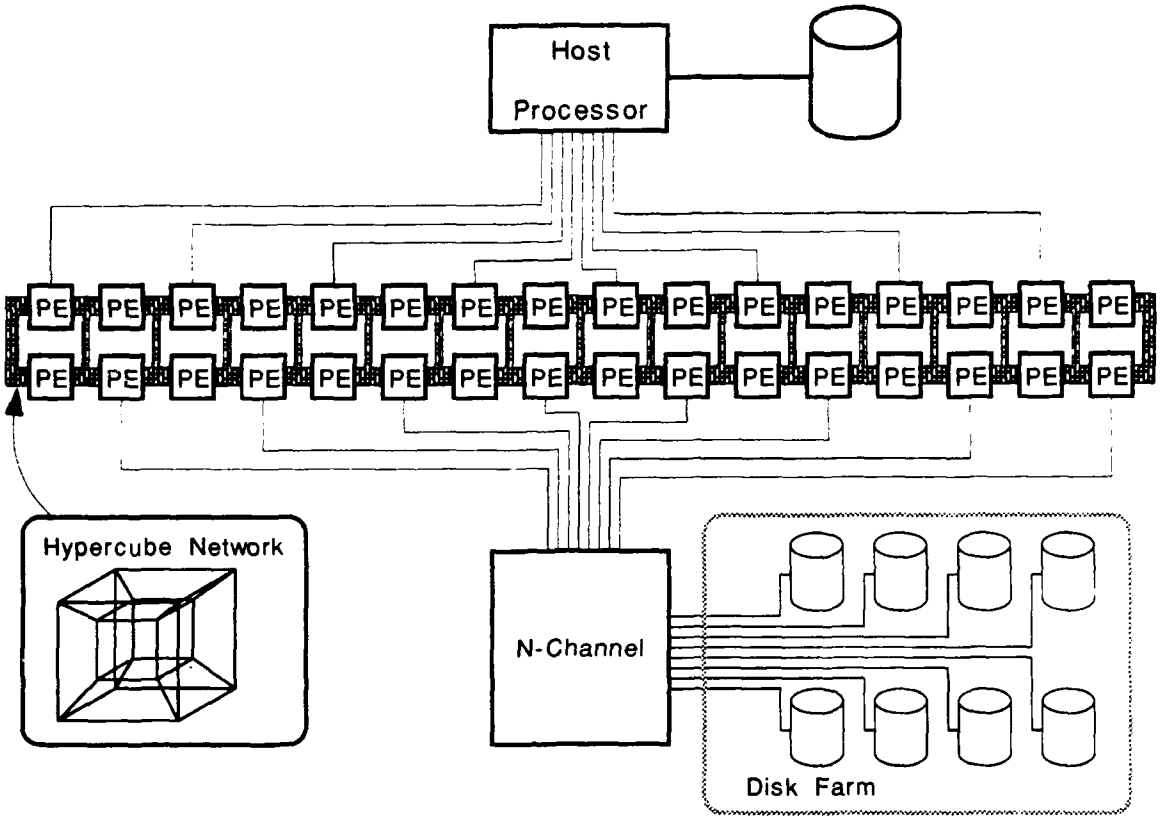

Figure 1. Schematic of an Ncube concurrent computer.

schematic of the Ncube hardware. An Ncube/seven can be configured to have up to 128 processors and it is a Multi-Instruction Multi-Data (MIMD) machine with local memory. All of the available memory is distributed among the processors and there is no shared memory. A pair of processors can access each others data through explicit message passing along communication channels connecting the pair. The Hypercube configuration places a processor at each vertex of an n-dimensional cube with the processors connected together through communication channels that are along the edges of the cube. An n-dimensional hypercube has 2^n processors with each processor connected to n other processors. The nodes are numbered 0 through n and in each node the communication channels are numbered 0 through $ln(n)$ (natural logarithm of the number of nodes). Then nodes i and j have a direct communication link if and only if the binary representation of i and j differ by one and only one bit. Hence, the channel number, c, connecting nodes i and j satisfies $|i - j| = 2^c$. For example, node 6 and 4 with binary representation 110 and 100 respectively, communicate through channel 1. The hypercube inter-connection scheme is one of the most general networks proposed and provides a rich inter-processor communication network.

Each node of the Ncube consists of a proprietary, custom VLSI chip and 512 Kbyte of

memory. Each chip contains a processor that is similar in architecture to a Vax-11 780 with floating point accelerator, 11 bidirectional direct memory access communication channels, and an error-correcting memory interface. With this number of channels, the largest possible configuration is a 10 dimensional hypercube with 1024 processor nodes (i.e Ncube/ten).

In addition to the node processors and inter-processor communication network, the Ncube provides subsystems to enable parallel input output (I/O). The I/O subsystem may be in the form of a host board, an n-channel board interface to a disk farm, and a graphics board. The Ncube/seven used in this study has a host board and an n-channel board with two disks. Each processor in the Ncube has a communication channel that can be used to pass data to one of the I/O subsystems. Certain nodes communicate directly with the host processor while others can pass data to the disk farm through the n-channel board. The operating system keeps track of the hardware configuration of each machine. A processor not connected to an I/O subsystem accesses data on an I/O subsystem through calls to system utilities. These utilities then route the data through the hypercube network and bring it to the processor. Writing data out to an I/O subsystem is done in a similar manner through a different system utility.

Due to lack of shared memory, the most suitable algorithms for implementation on local memory machines, such as the Ncube, are those that can partitioned into tasks that are nearly independent. That is each task requires a small amount (less that 1%) of its data from rest of processors. This is the case for the above frequency domain fluid-structure interaction problem.

CONCURRENT IMPLEMENTATION OF SWEEPS

The concurrent implementation of SWEEPS was developed from a VAX/VMS version of the software that uses an out-of-core equation solver to compute solutions to complex systems of the form in (1). This version was ported to the SUN front end workstation of the Ncube, and modified to work in-core. This in-core version was further split into two independent programs; a "host" program, and a "node" program. Figure 2 presents a schematic of the concurrent SWEEPS. The first program runs on the host computer of the Ncube. It reads the user and database input to SWEEPS and down-loads the appropriate data to the individual processors of the Ncube. The node program performs the frequency dependent calculation and has three components, one to receive the input data from the host program, another to evaluate the solution for each of the frequencies it has been assigned, and the last to output the result to the host computer. This program contains the computation intensive part of the analysis. Once the node processors of the Ncube complete their computation the host program then gathers the data from the node processors for post-processing.

EXAMPLE PROBLEM AND TIMING RESULTS

The problem chosen to illustrate the concurrent implementation of SWEEPS is a simple ring-stiffened cylindrical shell with end plates as shown in Figure 3, the whole being

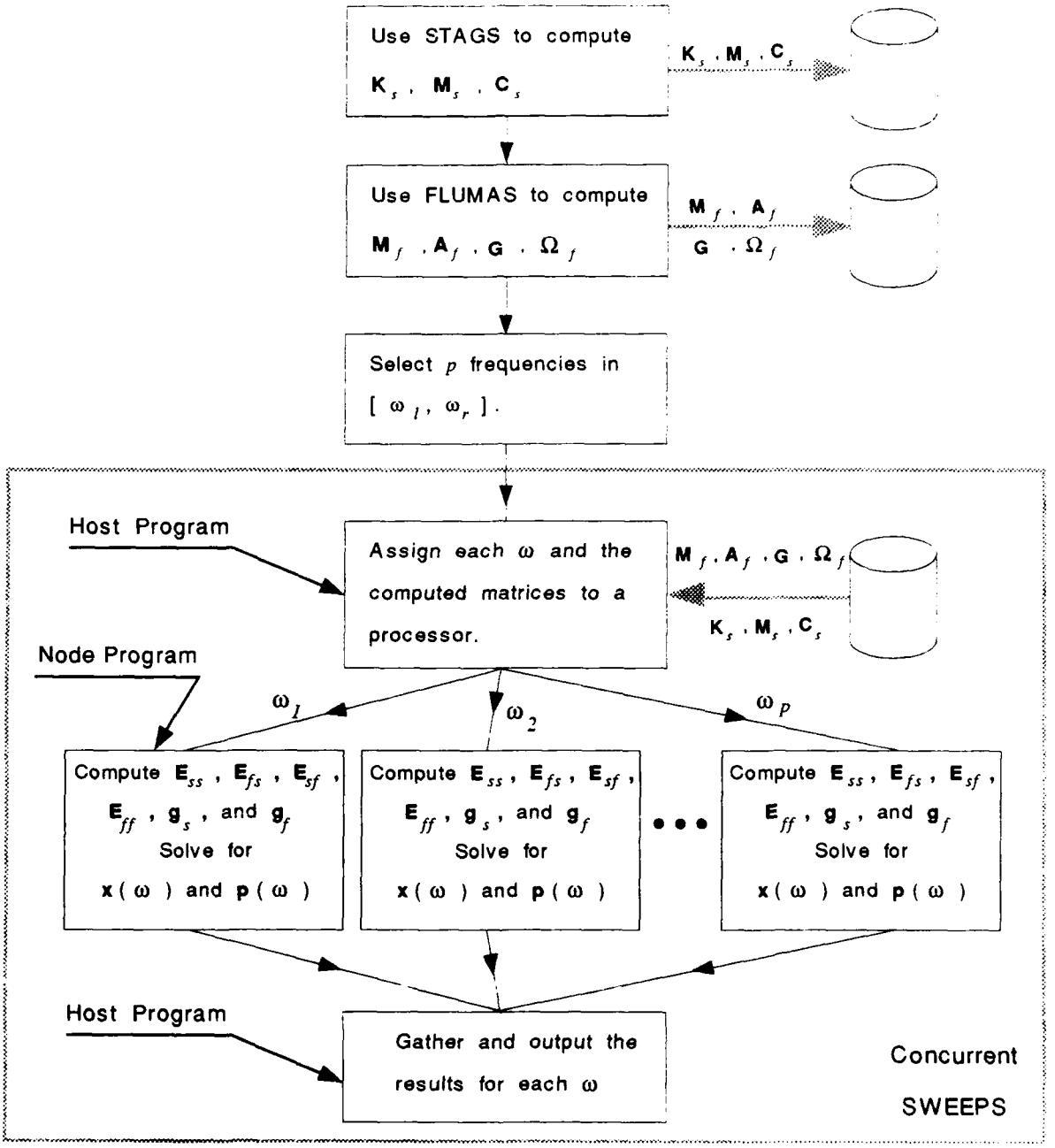

Figure 2. Architecture of concurrent SWEEPS.

made of steel. The shell is 60.75 inches long with an outside diameter of 40.5 inches and an 0.1875 inch wall thickness. The circular end plates are 1.50 inches thick with an

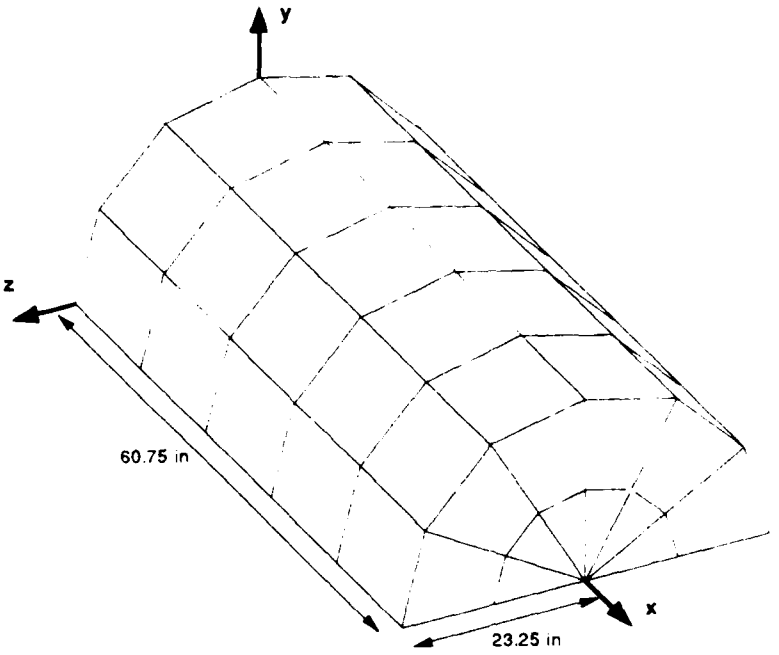

Figure 3. Finite element model of a ring stiffened cylindrical shell with end plates.

outside diameter of 46.5 inches. The shell stiffeners are of a "T" section construction with a web 1.1875 inches deep by .125 inches thick and a flange 1.03125 wide and 0.1875 inches thick. The stiffeners are equally spaced by 3.5625 inches in the axial direction of the shell.

A quarter model of the shell with an extremely crude finite element mesh was constructed with STAGS that contained 50 nodes and 42 elements and the stiffeners were simply smeared in the longitudinal direction, effectively making the shell orthotropic. The model contained 241 structural degrees-of-freedom and 57 fluid degrees-of freedom. For this problem 188 Kbytes of memory were required. With this implementation of concurrent SWEEPS, there is a total of 380 Kbyte of memory available. Based on this memory requirement it is anticipated that the largest possible problem that one can attempt on the Ncube is twice that of the cylinder problem.

The excitation consisted of a radially driven shaker at a point on the shell at mid length that covered the frequency range of 100 to 575 Hz and a constant loss factor of one percent was used over the entire frequency range. The quantity of most interest in this problem is the drive point mobility, i.e., the ratio of the radial velocity at the drive point to the amplitude of the driving force. The results of the computation are shown in Figure 4 and agree precisely with the CRAY X-MP and VAX 11 785 calculations. Note that the slight truncation of resonance peaks and anti-resonance valleys is due to the finite frequency increment of 5 Hz that was used.

Table 1 presents the computation time required to evaluate the drive point mobility at

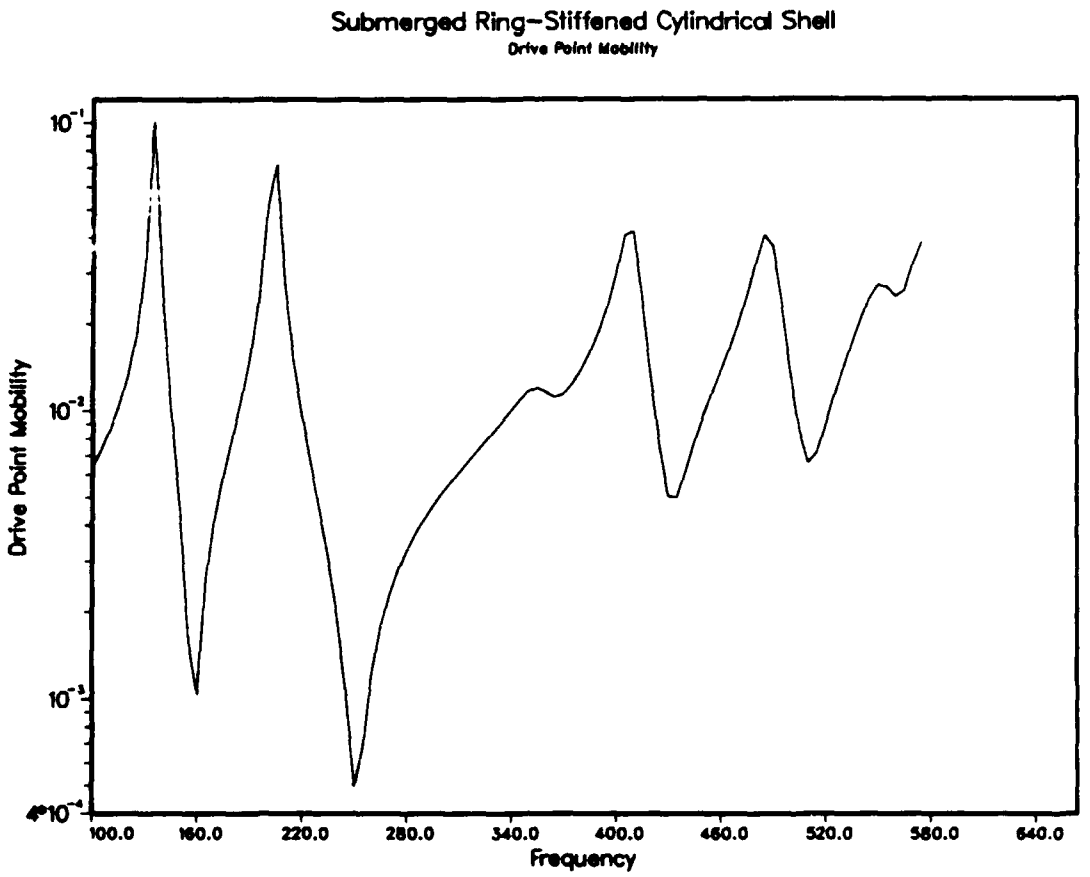

Figure 4. Drive Point Mobility for the submerged Ring-Stiffened Cylindrical Shell.

96 different frequencies. It demonstrates an almost linear speed up with the number of processors when using the Ncube computer. With 32 processors, the Ncube is slower than the Cray X-MP by a factor of five. In Table 2 a similar set of results is shown but for 256 frequency calculations. In this table the calculation time is given for both the slowest and the fastest processors of the Ncube. The difference between these two times (i.e. $t_{max} - t_{min}$) is a measure of the time that is required to output the results. Again, one can observe a linear speed up in the computation time (t_{min}) with the number of processors. With 256 processors trying to output their results, the single I/O device becomes the bottle neck. One of the processors will have to wait for the others to complete their job before it can output its results, thus becoming the slowest processor. This results indicates that without multiple disks (i.e. parallel I O), there is no gain in speed by using 256 processors instead of 128.

DISCUSSION AND CONCLUSIONS

The above results indicate that concurrent computation can be applied effectively to frequency domain calculations. This is especially true when the response of the system is required over a wide frequency range. Using the cylinder problem and an Ncube

Computer.	# of Processors	Solution time (sec.)
Vax 11/785	1	4896.0
Ncube	1	17861.8
Ncube	2	8930.9
Ncube	4	4465.7
Ncube	8	2233.1
Ncube	16	1116.5
Ncube	32	558.4
Cray X-MP	1	119.6

Table 1. Computer time for the evaluation of Drive Point Mobility at 96 frequencies.

Computer.	# of Processors	t_{min} (sec.)	t_{max} (sec.)
Ncube	32	1539.7	1595.1
Ncube	64	794.4	894.3
Ncube	128	424.8	622.7
Ncube	256	240.1	629.7
Cray X-MP	1	318.9	318.9

Table 2. Computer time for the evaluation of Drive Point Mobility at 256 frequencies.

computer, it was shown that the processing speed-up is roughly proportional to the number of processors in the concurrent computer. With 256 processors the computation time on the Ncube is less than that on the Cray X-MP.

However, as the number of processors increase, it was observed that the time required to output the results can dominate if parallel I/O devices are not used. Although the Ncube computer provides parallel I/O in the form of a disk farm, the implementation of the concurrent SWEEPS does not take advantage of this.

Further development of concurrent SWEEPS should focus on the issue of parallel I O by taking advantage of the disk farm of the Ncube. In addition, using an out-of-core equation solver, together with parallel I/O would overcome the memory limitation of the Ncube and thus one can model larger structures. It is also recommended that the latest version of SWEEPS be used for further development to take advantage of the exact external fluid formulation now in the code, as well as the ability to model internal fluid volumes [2].

ACKNOWLEDGMENTS

The work reported here was supported by the Office of Naval Research under contract N00014-82-C-0353 and, by the Lockheed Independent Research Program. The results presented in Table 2 were obtained using an Ncube/ten computer that was made available to Lockheed by the Ncube Corporation.

REFERENCES

1. J. A. DeRuntz, "Discrete Element Acoustic Analysis of Submerged Structures Using Doubly Asymptotic Approximations," LMSC–F035672, Lockheed Palo Alto Research Laboratory, Palo Alto, CA, April, 1985.

2. J. A. DeRuntz, I. C. Mathews, and M. A. Wright, "A Usage Primer for the Underwater Acoustic Radiation and Scattering Codes SWEEPS–ACRAD," LMSC-F301957, Lockheed Palo Alto Research Laboratory, Palo Alto, CA, March, 1989.

3. T. L. Geers, "Doubly Asymptotic Approximations for Transient Motions of Submerged Structures," *J. Acoust. Soc. Am.*, **Vol. 64, No. 5**, November, 1978, pp. 1500-1508.

4. H. A. Schenk, "Improved Integral Formulation for Acoustic Radiation Problems," *J. Acoust. Soc. Am.*, **Vol. 44, No. 1**, 1968, pp. 41-58.

5. G. Chertock, "Integral Equation Methods in Sound Radiation and Scattering from Arbitrary Surfaces," Report No. 3538, Naval Ship Research and Development Center, Carderock, Md., 1971.

6. I. C. Mathews, "Numerical Techniques for Three–Dimensional Steady–State Fluid–Structure Interaction," *J. Acoust. Soc. Am.*, **Vol. 79, No. 5**, May, 1986, pp. 1317-1325.

7. B. O. Almroth, F. A. Brogan, and G. M. Stanley, "Structural Analysis of General Shells, Volume II, User Instructions for STAGSC," LMSC-D633873, Lockheed Palo Alto Research Laboratory, Palo Alto, California, January 1980.

8. J. A. DeRuntz and T. L. Geers, "Added Mass Computation by the Boundary Integral Method," *Int. J. Num. Meth. Eng.*, **Vol. 12**, 1978, pp. 531-549.

9. J. A. DeRuntz, T. L. Geers, and C. A. Felippa, "The Underwater Shock Analysis Code (USA-Version 3), a Reference Manual", DNA 5615F, Defense Nuclear Agency, Washington, D.C., September, 1980.

10. C. A. Felippa, "Top-Down Derivation of Doubly Asymptotic Approximations for Structure-Fluid Interaction Analysis," *Innovative Numerical Analysis for the Engineering Sciences*, R. P. Shaw *et al.*, eds., University Press of Virginia, Charlottesville, 1980, pp. 79-88.

11. T. L. Geers and C. A. Felippa, "Doubly Asymptotic Approximations for Vibration Analysis of Submerged Structures," *J. Acoust. Soc. Am.*, **Vol. 73, No. 4**, April, 1983, pp. 1152-1159.